EECS 221- Internet of Things

Project Report

Reducing Energy Consumption via IoT and Machine Learning

Sudharshan Arutselvan (89979164), Ramkumar Rajabaskaran (85241493)

Introduction:

As summer is approaching, we can be sure that as temperatures soar, so will our electricity bill. This may be to various factors such as using more of air-conditioning, being more productive during summer, or even when we leave for summer holidays, we never quite fully turn off the power at our house. Thus there needs to be a method by which we should be able to reduce the electricity consumption in our houses as well as in our workplace.

Irvine is a geographical location in which during summer, the temperature has a high of 25-28C(80-86F) and a low of 13-16C(54-60F). We would like to have an ideal room-temperature of 19-21C (66-70F) . Hence by allowing the natural cooling effect of outside air, we can reduce the consumption of air-conditioners. Similar scenario can be stated for the use of lighting. Another way by which we can conserve energy is through turning off lights and fan, whenever we are not using or in the room (even if we forget) through the detection of human presence.

Thus by interfacing and fusing the data from various sensors and by applying concepts of machine learning, we can effectively reduce the amount of power consumed and thereby reduce the cost of electricity. This not only proves useful for the user, but also to the environment.

Project Goal:

The aim of the project is to create a smart home system that can sense, monitor and control the temperature in a particular room. The comfort zone of the residents need to be observed and create a learning system that can provide efficient ways to keep the temperature in the comfort zone and reduce the energy consumption.

Related Works:

The energy cost management of HVAC systems has recently attracted the research attention. In [1], the energy cost is studied as a function of the parameters that control the air and water subsystems and an evolutionary programming method is proposed to save energy. Moreover, in [2], a dynamic threshold controls the energy consumption and it varies according to the user satisfaction, which also depends on a thermal model. However, neither [1] nor [2] explicitly consider a dynamic pricing cost. In [3], smart pricing is considered in the energy cost optimization, but the user comfort is not explicitly incorporated in the algorithm, as the authors consider that the HVAC is turned on/off when the indoor temperature is outside the margin of comfort. Recently, in [4, 5], both energy scheduling of HVAC under smart pricing and the user comfort are considered. In [4], Nguyen et al. propose the construction of a lookup table of room temperatures that depends on (i) the past temperatures, (ii) the outdoor temperature, and (iii) the HVAC power. The authors claim that the lookup table is built during a training period (that takes place only once) and permits to assess the temperature of comfort for a given operation of the HVAC energy scheduler. However, this heuristic approach seems hardly applicable in general scenarios. In [5], a linear energy cost function is considered, although quadratic or two-step piecewise linear functions are more common in practice [6], while user's comfort is measured only at a specific location. It is also worth noting that none of the works considers an IoT framework.

Unlike [1-3], in our proposed energy scheduling methods, both the smart pricing tariffs and the user comfort are considered. Moreover, the temperature of comfort is measured at several building positions by different sensor nodes that form a wireless sensor network (WSN), thus providing a more accurate measure of comfort compared to [5]. Furthermore, compared to [4], we adopt a more analytical and less heuristic model to assess the user comfort in the HVAC energy cost optimization, our model considers the time varying nature of the thermal conditions, without requiring a training period. Moreover, our model adaptively updates the past temperature measurements for each time, whereas the model in [4] is only carried out once to construct the lookup table for indoor and outdoor conditions. Finally, unlike most of the above references, our methods are validated in a real scenario.

Architecture:

The project has three parts,

- Sensing.
- User Interface
- Data analytics and decision.
- Controlling.

The sensors collect the information about the environment in the rooms and feed this data along with number of occupants in the room and time of the day to the machine learning environment. The data is stored and used to learn the comfort zone of the user at each intervals of the day and year. The set then makes a decision based on previous results on the temperature needed in the room and the control action needed to achieve that. This is sent to the control device which make suitable actions.

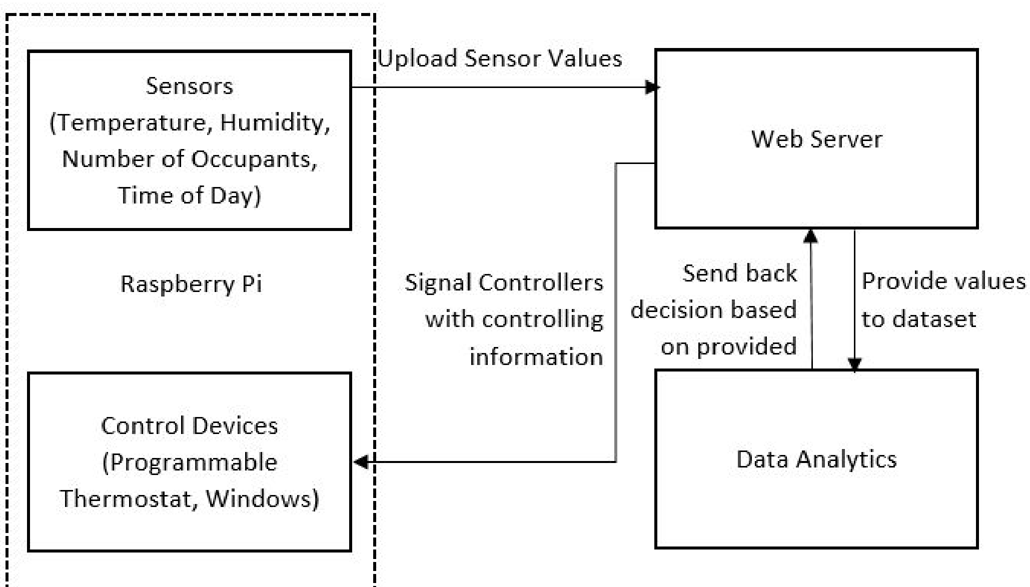

Fig 1. Architecture of the proposed system.

Sensors:

Current sensor:

- SMAKN® ACS712 Current Sensor

Temperature, Pressure and Humidity module

- Adafruit BME280

Distance Sensor:

- SHARP GP2Y0A21YK, an InfraRed distance sensor.

Force Sensitive Resistor (FSR):

- A sensor to vary resistance based on pressure applied. This is used along with distance sensor to detect user entry or exit from the room.

The sensors mentioned above are used in unison. The current sensor is used to detect whether the device is on or off and senses the current consumption. With this information we will be able to calculate the total power consumed by the device for a particular time and store this information in a database. This information is then taken to create the statistics on how the consumption is and we can compare with normal consumption by the device to check the conservation of energy. The sensor arrangement in real time is as in the figure below.

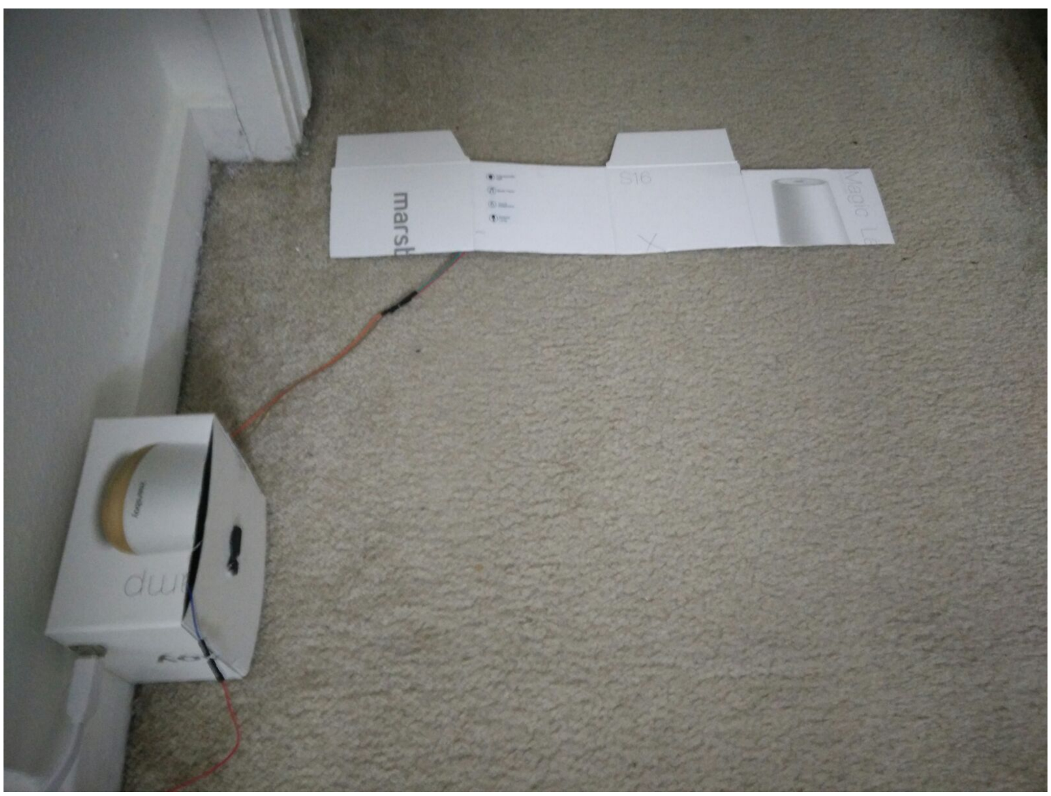

Figure 2: Motion Detector Setup.

The temperature, pressure and humidity sensor are used to sense the temperature and humidity inside the room, which will be sent to the Azure Machine Learning service to get the information about the comfort zone of a particular environment.

The distance sensor and force sensitive resistor are bound together to determine whether a user is entering the room or not. The sensors work together. The pressure sensor will be placed near the entrance of the room and the distance sensor a little inside the room. Whenever a person tries to enter, they place their foot on the force sensitive resistor which will detect human movement and once they cross the distance sensor which detects the presence, then the sensors will decide that the user has entered the room. If the reverse of this happens, then the sensors decide the person is leaving the room. This helps in keeping count of number of users presently in the room. Hence the system will only work if any person is in the house and becomes idle in absence. This is one part of energy conservation.

All these data are collected together by the raspberry pi and formatted properly and sent to the cloud service through the web server. Additionally the light sensor is used to determine whether or not the user will need the light inside the room and sends information in similar fashion as mentioned above.

User Interface:

The user interface provides features like showing the current room temperature, the temperature predicted, weather forecast, energy consumed and also provides interface to change the temperature predicted. This interface looks like a thermostat which lets the user control the temperature. The Home Page with a Knob to adjust the Room temperature. Currently the Temperature is Set at 80 °F. There are also Two buttons that take the user to the Weather Page and the Energy page. The Weather page provides the user with the current Temperature and Weather Forecast Data. The Energy Page provides the user with the statistics such as Energy consumption. The user interface for the application is as below:

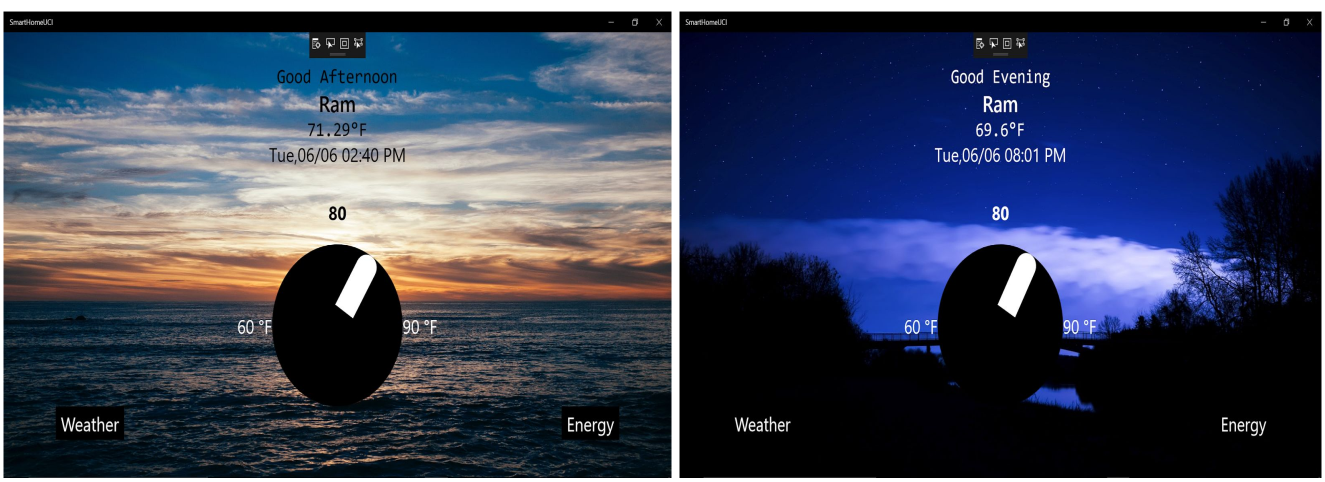

Figure 3:UI during (a) Afternoon and (b) Night

Data Analytics and Decision:

The information received from the sensors are sent to the Azure machine learning service. The data will be used to check with previous results in the dataset and create a decision for the air conditioning system.

Cortana is Microsoft's fully managed big data and advanced analytics suite. It includes four main components: Information Management, Big Data Stores, Machine Learning and Analytics, and Dashboard and Visualization. It has the following features.

- Information Management: where data is collected from different resources.
- Big data stores include SQL Database, Azure SQL Data Warehouse and Azure Data.
- Machine Learning and Analytics includes machine learning and big data modules.
  - It has various machine-learning API's which can be utilized to easily implement Machine learning aspect in our project.
- Dashboard and Visualization different ways to understand, present and visualise data.

We plan to utilize these features collectively to provide a smart IoT solution for our project.

The aim of the machine learning in this project is to create a comfort zone for the resident inside the rooms. Residents feel comfortable in a temperature range only. This needs to be modeled and calculated in order to keep the resident in their comfort zone. They need not keeping changing the temperature. The system will be able to decide for itself from the previous data and create the comfort zone. Whenever a particular temperature is queried it determines the nearest comfort zone and sends it as prediction to keep the room in that temperature. This can be done by using the voting system.

Predicted Mean Vote:

The vote here is the temperature set by the user and the control. The IoT system sets a comfort temperature for the user and works on it. When the user feels the temperature is not optimal or comforting, the user sets a different temperature. This is called voting against the temperature set. Hence, the model needs to make changes to its mean to keep the system in a particular temperature suitable for user. The model has already calculated a mean vote with previous data hence, the newly voted temperature is sent as feedback and the predicted mean vote is changed to adaptive predicted mean vote. These are represented as equations in the following figure.

$$aPMV = \frac{PMV}{1 + \lambda * PMV}$$

$$\lambda = Adaptive\ coefficient = \frac{1}{Temperature\ Set}$$

$$PMV = Predicted\ Mean\ Vote$$

$$aPMV = Adaptive\ Predicted\ Mean\ Vote$$

Figure 4: Equation for machine learning algorithm.

The machine learning in Azure Machine learning service provided by Cortana is used by building the different blocks needed for it.

The following figure shows the built machine learning service for this application.

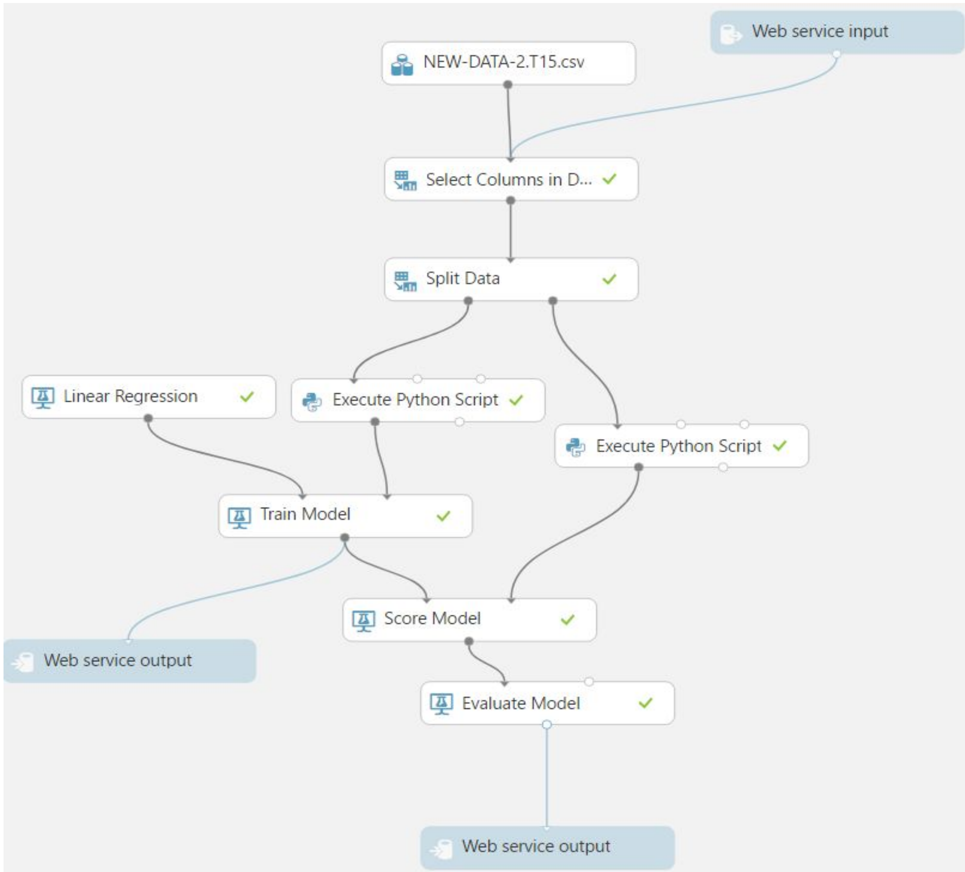

Fig 5. The model of the machine learning algorithm

Steps to build the machine learning service:

- Initially the dataset has to be uploaded and a new dataset has to be created. The dataset used for this project was taken from UCI dataset.
- Once the dataset is prepared it can be imported. Next, this needs to be cleaned up, since there might be few data which are not proper or might have null values. These are next cleaned up.
- From the cleaned up data we will be needing only a few columns for machine learning and hence use a select column block and change the parameters to select only those columns needed.
- For the purpose of scoring and evaluating, the model we get out of machine learning we need a set of information from this dataset. Hence, the data is split into 70:30 percentage and 70% is used for machine learning and remaining 30% is used for scoring and evaluation.
- For the purpose of the calculating the adaptive predicted mean vote, a python script is available which will use the equation explained above for predicted mean vote and calculate the adaptive vote.
- The linear regression is used to calculate the mean of the total dataset for a few columns and then use them in the equations.

- We have the predicted mean vote calculated. Now this has to be used to train the model on how the model will be for the future inputs. Hence, the train model block is used to train our machine learning set.
- Once the training of the set is done, the model needs to be created to scored with previous dataset and check whether they are close to what was needed. The scoring model block does this function and using this we will be able to get the output for our temperature model.
- Next the evaluation model is for the training purpose as it provides analysis of how close the learning is to the actual data.
- Once all these steps are done. The machine learning model is run for few iterations to train it well.
- These steps have perfectly created a good machine learning module. Now this needs to be plugged to a web service so that the application will be able to query and get response on the temperature data. Hence, the module is created as a predicted model and then attached to the web service input and output. The web service will provide url to send the needed temperature and humidity data to this machine learning service and respond back with the comfort temperature data.

Control Devices:

The electrical appliances can be controlled using relays. The air conditioner cannot be controlled only by relays. Relays will control their switching on or off but the system should be able to control the temperature that the air conditioner should work at.
Nest: Smart thermostat . In the case of a centralised Heating / Cooling system, In many houses, we can build a smart thermostat to override the existing thermostat , like NEST , that can be controlled through a mobile app or can be made to learn and reduce energy consumption through machine learning.

Connection:

Raspberry pi will be used to connect all the devices that sense data. The data needs to be sent to the cloud servers to make computations. This can be achieved through several ways and we have decided to use web servers to communicate between the Raspberry pi and the cloud service. The web server service is provided by the Azure platform itself and it can be built according to the project need. We are using HTTP Requests to send and receive data from the Raspberry Pi and the Web Service.

We have wired the ADC , BME280 using I2C Communication. The Relay module is controlled using GPIO pins. The Current Sensor, Force Sensor and Distance Sensor are read using the ADC (Analog to Digital converter), which is a 16-bit Precision 4 Channel ADC.

The Inter-integrated Circuit (I2C) Protocol is a protocol intended to allow multiple “slave” digital integrated circuits (“chips”) to communicate with one or more “master” chips. Like the Serial Peripheral Interface (SPI), it is only intended for short distance communications within a single device. Like Asynchronous Serial Interfaces (such as RS-232 or UARTs), it only requires two signal wires to exchange information. The Inter integrated Circuit (I2C) Protocol is a protocol intended to allow multiple “slave” digital integrated circuits (“chips”) to communicate with one or more “master” chips. Like the Serial Peripheral Interface (SPI), it is only intended for short distance communications within a single device. Like Asynchronous Serial Interfaces (such as RS-232 or UARTs), it only requires two signal wires to exchange information. I2C requires a mere two wires, like asynchronous serial, but those two wires can support up to 1008 slave devices. Also, unlike SPI, I2C can support a multi-master system, allowing more than one master to communicate with all devices on the bus.

Data rates fall between asynchronous serial and SPI; most I2C devices can communicate at 100 kHz or 400 kHz. There is some overhead with I2C; for every 8 bits of data to be sent, one extra bit of Meta data (the “ACK/NACK” bit) must be transmitted. The hardware required to implement I2C is more complex than SPI, but less than asynchronous serial. It can be fairly trivially implemented in software.

Results and Experiments:

Result of Machine Learning:

Mean Absolute Error	0.985591
Root Mean Squared Error	1.251093
Relative Absolute Error	0.477153
Relative Squared Error	0.261162
Coefficient of Determination	0.738838

These values say that how close the predicted value is to the expected value. The mean absolute error shows the difference could go to a mean of almost 1 degree. The coefficient of determination shows that the model represented here is almost 75% predictable. It is not 25% because, the data set provided is only for summer season and not other seasons, hence if we get a dataset for a year, the system will be able to predict with perfection.

Hence, the machine learning model for the given dataset is providing good results in predicting the optimal temperature for the comfort zone of the resident.

We ran the experiment for about half a day and we received few results. We calculated the power consumed by the device and created a plot for that. We also considered the time it took for a process and calculated the estimated power consumed if only air conditioner is used. We were able to get few results as shown below. We have used a fan as the cooling device here instead of the air conditioner. Hence, the results shown below are for the fan controlled by our IoT based thermostat.

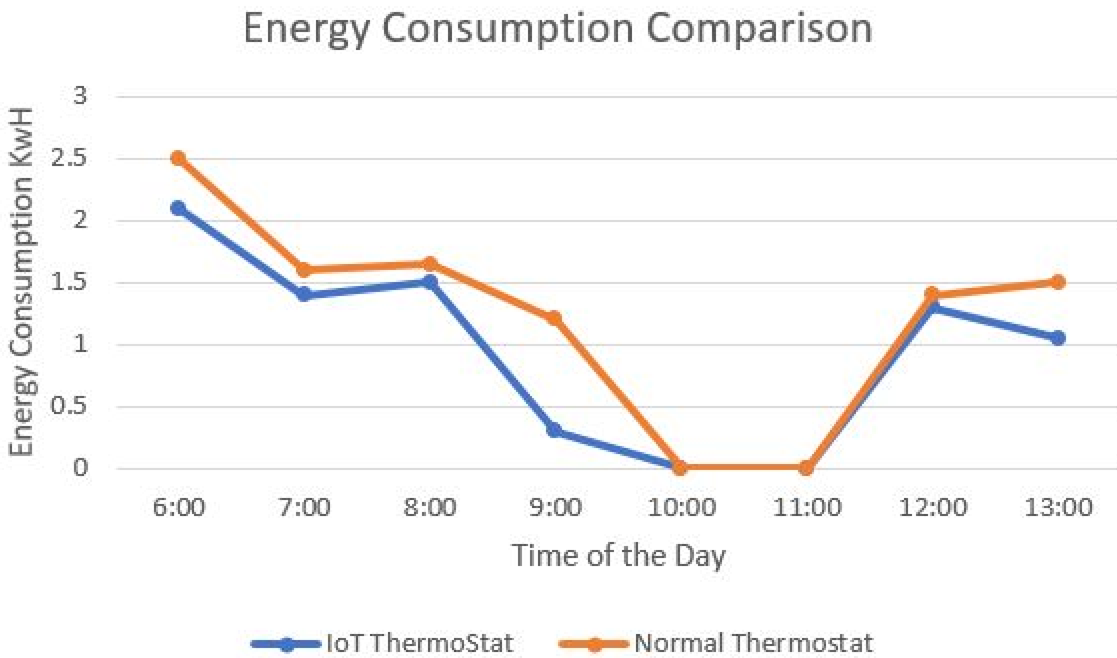

Figure 6: Comparison of energy consumption between the IoT based Thermostat created to normal thermostats.

From the graph it is clear that the control used by the IoT based Thermostat is more efficient than the normal thermostats. The explanation for the graph is that, we started taking reading from morning 6 AM till afternoon 1 PM and observed the above result.

Early morning the temperature needed by the user was higher than room temperature and hence the fan had to run most of the time. Whereas by 10AM to 11 AM no user was available in the house meaning hey did not need any control, hence both gives zero consumption. But as you can see once user enters, both the thermostat gives similar result because there will be constant running rather than periodic which is similar to normal thermostat.

Conclusion:

The IoT system to reduce the energy conservation by controlling the devices in the house using machine learning has been completed. Machine learning was implemented to predict comfort zone of the residents in a particular room which was successful. The machine learning algorithm used was predicted mean vote which will be able to predict the comfort zone using the voting system. The algorithm learns as it is used for longer. Using this information, the devices were controlled to conserve energy. Energy conservation is done by using a collective actuators. Instead of using air conditioner all the time to reduce or increase to a particular temperature, the system determine environment settings and will control the windows and fans. Hence by reducing the number of hours of running of the air conditioner, the energy has been conserved.

References:

[1] Fong KF, Hanby VI, Chow TT. HVAC system optimization for energy management by evolutionary programming. *Energy and Buildings*. 2006;38(3):220–231.

[2] Ha DL, de Lamotte FF, Huynh Q-H. Real-time dynamic multilevel optimization for demand-side load management. Proceedings of the IEEE International Conference on Industrial Engineering and Engineering Management (IEEM ’07); December 2007; Singapore. pp. 945–949.

[3] Noh S, Yun J, Kim K. An efficient building air conditioning system control under real-time pricing. Proceedings of the International Conference on Advanced Power System Automation and Protection (APAP ’11); October 2011; Beijing, China. pp. 1283–1286.

[4] Nguyen HT, Nguyen D, Le LB. Home energy management with generic thermal dynamics and user temperature preference. Proceedings of the IEEE International Conference on Smart Grid Communications (SmartGridComm ’13); October 2013; Vancouver, Canada. pp. 552–557.

[5] Avci M, Erkoc M, Asfour SS. Residential HVAC load control strategy in real-time electricity pricing environment. Proceedings of the IEEE Energytech; May 2012; Cleveland, Ohio, USA.

[6] Mohsenian-Rad A-H, Wong VWS, Jatskevich J, Schober R, Leon-Garcia A. Autonomous demand-side management based on game-theoretic energy consumption scheduling for the future smart grid. *IEEE Transactions on Smart Grid*. 2010;1(3):320–331.